Journal of KIIT. Vol. 18, No. 10, pp. 1-8. Oct. 31. 2020. pISSN 1598-8619. eISSN 2093-7571
http://dx.doi.org/10.14801/jkiit.2020.18.10.1

SDN에서 강화학습을 통한 효과적인 멀티캐스트 라우팅 트리 생성 방법

채지훈*, 이병대**[1], 김남기**[2]

An Efficient Multicast Routing Tree Construction Method with Reinforcement Learning in SDN

Jihun Chae*, Byoung-Dai Lee**[1], and Namgi Kim**[2]

본 논문은 2020학년도 경기대학교 대학원 연구원장학생 장학금 지원에 의하여 수행되었음. 또한 교육부의 재원으로 한국연구재단의 지원(NRF-2017R1D1A1B04027874)을 받아 수행되었음.

요 약

인공지능 기술의 발전과 함께 네트워크 분야에서 라우팅 문제에 대해 강화학습(Reinforcement Learning)을 적용하는 연구들이 속속 등장하고 있다. 하지만 기본적인 강화학습 방법은 고정적인 환경을 가정하기 때문에 시간에 따라 변화하는 가변적인 네트워크 환경에서 성능에 한계를 가진다. 본 논문에서는 이러한 한계를 극복하고 SDN에서 가변적인 네트워크 환경을 잘 반영할 수 있는 심층 강화학습 기반의 멀티캐스트 라우팅 트리 생성 방법을 제안한다. 본 논문에서 제안하는 방법을 평가하기 위해 다양한 네트워크 토폴로지에서 성능을 비교하는 실험을 진행하였다. 그 결과 네트워크 토폴로지가 고정된 환경에서 학습한 강화학습 에이전트보다 제안 방법으로 학습한 강화학습 에이전트가 다양한 네트워크 토폴로지에서도 보다 최적에 가까운 멀티캐스트 라우팅 트리를 생성함을 알 수 있었다.

Abstract

Along with the development of artificial intelligence technology, researches that apply reinforcement learning to routing problems in the network field are emerging. However, the basic reinforcement learning method assumes a fixed environment, so performance is limited in variable network environment that varies over time. Therefore, we proposes a deep reinforcement learning-based multicast routing tree construction method that can overcome these limitations and reflect the variable network environment in SDN. To evaluate the method proposed, experiments were performed to compare performance in various network topology. As a result, It was found that the deep reinforcement learning agent learned by proposed method in various network topology produced optimal close multicast routing tree than deep reinforcement learning agent learned in fixed network topology.

Keywords

Artificial Intelligence(AI), Software Defined Networking(SDN), deep reinforcement learning, multicast routing

* 경기대학교 컴퓨터공학부 석사과정
- ORCID: https://orcid.org/0000-0001-8040-0693
** 경기대학교 컴퓨터공학부 교수([2]교신저자)
- ORCID[1]: https://orcid.org/0000-0002-4028-6168
- ORCID[2]: https://orcid.org/0000-0002-0077-6576

· Received: Aug. 14, 2020, Revised: Oct. 04, 2020, Accepted: Oct. 07, 2020
· Corresponding Author: Namgi Kim
Division of Computer Science and Engineering, Kyonggi University, 154-42, Gwanggyosan-ro, Yeongtong-gu, Suwon-si, Gyounggi-do, Korea
Tel.: +82-31-249-9962, Email: ngkim@kyonggi.ac.kr

Ⅰ. 서 론

네트워크 기술의 발전에 따라 다양한 서비스들이 생겨나고 있고 이로 인해 네트워크에서 처리해야 하는 데이터의 양이 급속도로 증가하고 있다. 이에 따라 대량의 데이터를 다룰 수 있는 인공지능 및 머신러닝 기술을 활용하고자 하는 연구들이 활발히 진행되고 있다[1][2]. 그 중에서도 강화학습은 네트워크의 최적화, 관리 및 운영 등의 자동화와 같은 분야에서 주목받고 있다. 동시에 기존의 폐쇄적인 분산 네트워크에서 벗어나 SDN(Software Defined Networking) 기술과 같은 새로운 네트워크 패러다임에서의 연구 또한 활발히 진행되고 있다. 이에 따라 SDN 환경에서의 문제들에 강화학습 방법을 적용한 연구들이 진행되었으나 가변적인 네트워크에서 그 성능에 한계가 존재하며 안정적이지 않았다[3]. 따라서 본 논문에서는 이러한 문제를 해결하기 위해 기존의 고정적 환경을 가정하여 설계된 강화학습 방법과는 다른 학습 방법을 제안하고 실험을 통해 그 성능을 평가한다.

Ⅱ. 관련 연구

2.1 SDN에서 멀티캐스트 라우팅

네트워크에서 패킷을 목적지까지 전송하기 위해서는 유니캐스트, 멀티캐스트, 브로드캐스트 등의 방법이 존재한다. 유니캐스트 라우팅 기법은 패킷을 전송해야 할 경우 목적지 수만큼 패킷을 출발지에서 생성해서 전송해야 한다. 반면에 멀티캐스트 라우팅 기법은 패킷 전송 시 네트워크의 라우터가 패킷을 복사하여 목적지로 패킷을 보내기 때문에 유니캐스트 라우팅 기법에 비해서 효율적으로 같은 데이터를 여러 목적지로 전달할 수 있다. 그러나 이러한 멀티캐스트 라우팅 기법은 라우팅 시에 멀티캐스트 라우팅 트리를 형성하게 되는데 각 라우터가 독립적으로 운영되는 기존의 분산 네트워크에서는 흩어진 네트워크 연결 정보로 인해 최적인 멀티캐스트 라우팅 트리를 형성하는 것이 쉽지 않다. 하지만 기존의 분산 네트워크 환경과 다르게 SDN 환경은 네트워크 제어 평면과 데이터 평면으로 분리하였기 때문에 SDN 제어기(Controller)가 제어 평면을 이용해 네트워크를 중앙집중적으로 제어하는 것이 가능하다. 이러한 장점으로 SDN 환경에서 멀티캐스트 라우팅 트리는 기존의 분산 네트워크 환경에서보다 효과적으로 형성할 수 있다. 따라서 SDN 환경에서 멀티캐스트 라우팅 트리를 보다 효율적으로 생성하는 연구들이 활발히 진행되고 있다[4]. 그런데 SDN 환경에서 멀티캐스트 라우팅 트리를 생성하는 문제는 대게 스타이너 트리(Steiner Tree)를 생성하는 문제로 귀결되고 이는 NP-Complete의 문제이다[5][6].

2.2 강화학습

강화학습은 기본적으로 그림 1과 같은 구조를 가진다. 그림에서 에이전트(Agent)는 환경(Environment)과 상호작용하면서 환경에서 인식한 현재 상태에서 수집한 경험으로부터 가능한 행동(Action)을 선택한다. 또한 강화학습은 보상(Reward) 함수에 기반을 두고 있는데 보상은 환경으로부터 에이전트가 전달받는 것으로 현재 취한 행동에 대한 피드백을 받는 것으로 볼 수 있다. 전통적인 강화학습 방법에는 Q-learning[7], SARSA[8]와 같은 기법들이 존재하나 이 기법들은 여러 한계점을 가지고 있다. 그 중 하나는 모두 테이블 기반의 방법론으로서 에이전트와 환경이 상호작용하면서 이루어질 수 있는 모든 상태와 행동의 경우에 대해 사전에 기록하여 최대의 보상을 얻도록 하는 방식을 사용한다는 것에 있다. 이러한 테이블 기반의 방법론은 해결하고자 하는 문제가 복잡한 문제일 경우 무한히 많은 상태와 행동 쌍에 대한 경우를 저장할 수 없기 때문에 계산 복잡도 및 차원의 저주 같은 문제점이 있다. 따라서 이를 해결하기 위해 Deep-Q-Network(DQN)과 같이 인공신경망을 이용하는 방법이 제안되었다[9].

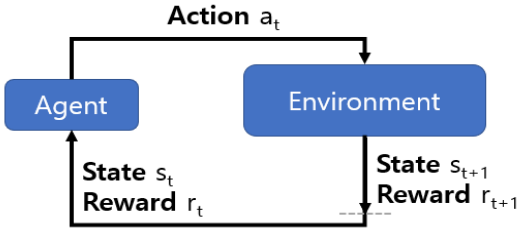


그림 1. 강화 학습 구조
Fig. 1. Reinforcement learning model

따라서 DQN 기법을 여러 분야의 적용하는 연구들이 진행되었는데 [10]은 멀티캐스트 라우팅 트리 생성 문제에 DQN 기법을 적용한 연구이다. [10]에서는 학습과 테스트 모두 동일한 네트워크 토폴로지를 가정하고 멀티캐스트 노드 케이스에 대한 변화를 고려한 실험을 진행하였다. 그러나 네트워크 환경은 링크 연결 정보가 수정될 수 있는 가변적 특징을 가지고 있기 때문에 가변적 네트워크 링크 토폴로지에서는 적절한 성능을 나타내지 못한다.

따라서 본 논문에서는 멀티캐스트 노드 케이스 뿐만 아니라 다양한 링크 토폴로지 케이스를 사용하여 가변적인 네트워크 링크 토폴로지에서 유연하게 멀티캐스트 라우팅 트리를 생성하는 학습 방법을 제안한다.

III. 제안 기법

3.1 SDN에서 멀티캐스트 라우팅 트리 생성 문제 정의

SDN 환경에서 멀티캐스트 라우팅 트리 생성 문제는 다음과 같이 정의할 수 있다. SDN 환경에서 네트워크를 구성하는 노드들은 SDN 제어기에 의해서 완전하게 통제되고 관측 가능하다. 따라서 기존의 분산 네트워크와 다르게 전체 네트워크 관점에서 라우팅이 가능하다. 이때 전체 네트워크 관점에서의 라우팅은 출발 노드에서 모든 목적지 노드들까지 최소 비용으로 연결하는 문제이다. 이러한 문제는 스타이너 트리를 생성하는 문제로 알려져 있고 대게 NP-Complete의 문제로 귀결된다. 따라서 네트워크를 구성하는 노드의 수가 증가할수록 최적의 스타이너 트리를 찾는 것을 어렵다.

따라서 본 연구에서는 인공지능을 이용한 접근 방법으로서 강화학습을 이용하여 멀티캐스트 라우팅 트리 생성 문제를 다룬다. 또한 고정적 네트워크 토폴로지 환경을 가정하지 않고 가변적인 네트워크 토폴로지 환경을 가정하여 SDN에서 멀티캐스트 라우팅 트리를 생성하고자 한다. 다음은 강화학습 기법을 이용하기 위해서 SDN에서 멀티캐스트 라우팅 트리 생성 문제를 MDP(Markov Decision Process)로 나타낸 것이다. MDP M은 $M = \{S, A, R, \gamma\}$의 집합으로 구성된다. S는 상태(State)를 나타내며 강화학습 에이전트가 환경으로부터 관측한 정보를 나타낸다. SDN에서는 제어기가 강화학습 에이전트의 역할을 하므로 SDN에서 발생하는 네트워크 지연, 대역폭, 링크 정보 등의 모든 정보를 상태로 관측하는 것이 가능하다. A는 행동(Action)으로 강화학습 에이전트가 관측한 상태로부터 행동 가능한 행동 집합을 나타낸다. 즉, 멀티캐스트 라우팅 트리를 생성하는 문제에서 행동은 다음으로 연결 가능한 링크들이 된다. R은 보상(Reward)을 나타낸다. 보상은 강화학습 에이전트가 현재 상태에서 어떤 행동을 했을 때 변화한 환경으로부터 주어지는 값이다. 따라서 멀티캐스트 라우팅 트리 생성 문제에서는 목적지 노드에 연결되면 보상이 발생한다. γ는 0~1 사이의 값을 가지며 미래의 보상에 대한 감가율을 나타낸다. 보상은 강화학습 에이전트가 적은 행동을 하면서 목표에 도달할수록 누적되어야 한다. 따라서 멀티캐스트 라우팅 트리 생성 시에 적은 링크 개수를 선택할수록 누적 보상이 증가된다.

3.2 학습 방법

그림 2는 학습된 DQN 에이전트를 통해 멀티캐스트 라우팅 트리를 생성하고 이를 바탕으로 플로우 규칙(Flow Rule)을 SDN 스위치들에게 전달하는 과정을 보여주고 있다. SDN 환경에서 SDN 제어기는 여러 SDN 스위치로부터 네트워크 상황을 측정한 여러 정보들을 전달 받는다.

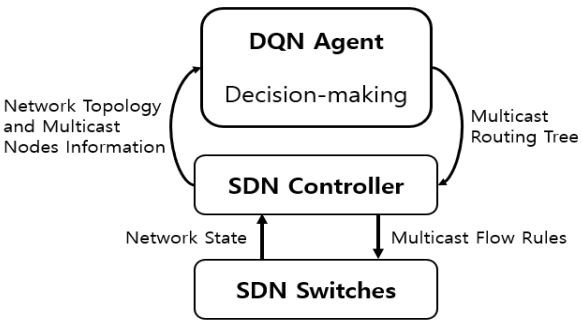


그림 2. DQN 에이전트 기반의 멀티캐스트 라우팅 트리를 생성하고 이를 바탕으로 플로우 규칙으로 전달하는 과정

Fig. 2. Process of forwarding the flow rule using multicast routing tree constructed based on DQN agent

SDN 제어기는 전달 받는 정보들 중에서 멀티캐스트 출발지, 목적지들에 대한 정보와 현재 네트워크 토폴로지의 링크 연결 정보를 DQN 에이전트에게 전달한다. 효율적인 멀티캐스트 라우팅 트리를 생성하도록 학습된 DQN 에이전트는 전달받은 정보를 이용하여 해당 네트워크 상황에 맞는 멀티캐스트 라우팅 트리를 생성하여 SDN 제어기에게 전달한다. SDN 제어기는 멀티캐스트 라우팅 트리에 따라 플로우 규칙을 생성하여 SDN 스위치들에게 전송한다. 그 후 출발지 노드의 멀티캐스트 패킷은 각 스위치에 존재하는 플로우 규칙에 따라 멀티캐스트 라우팅되어 목적지에 도착한다.

그림 3은 DQN 에이전트를 학습시키는 과정을 보여주고 있다. 그림에서 DQN 에이전트는 SDN 제어기로부터 네트워크 정보 즉, 상태 S를 전달받는데 여기에는 멀티캐스트 요청, 링크 연결 정보, 지연 등과 같이 스위치에서 제어기로 수집된 정보가 포함된다. DQN 에이전트는 이러한 상태 S에서 다음 연결할 노드를 e-greedy 방법을 이용하여 결정한다. e-greedy 방법을 이용할 때, 연결 가능한 다음 행동들 중에서 무작위로 선택하거나 신경망(Neural Network)에 입력으로 전달하여 나온 결과로부터 최대 Q-value를 가지는 행동을 선택한다. 이렇게 선택된 행동, 즉 노드가 목표로 하는 멀티캐스트 요청의 목적지 중 하나이면 보상 값을 환경으로부터 전달받는다. 이러한 과정에서 발생하는 경험들을 리플라이 버퍼에 저장하고 모든 목적지를 찾게 되면 한 번의 에피소드(Episode)가 종료된다.

이때 본 논문에서 제안하는 방법은 한 번의 에피소드가 종료될 때 마다 생성 가능한 네트워크 링크 토폴로지의 분포로부터 새로운 네트워크 링크 토폴로지 환경을 추출한다. 따라서 기존 연구와 달리 DQN 에이전트는 에피소드마다 새로운 네트워크 상황에서 학습하게 된다. 새로운 환경에서 DQN 에이전트가 일정한 수의 에피소드만큼 경험하게 되면 리플라이 버퍼에 저장되어 있던 경험들에서 미니배치 데이터를 무작위로 추출한다. 이 데이터를 이용해 학습 신경망을 학습 시킨다. 또한 일정 에피소드마다 목적 신경망으로 학습 신경망의 가중치을 복사한다.

그림 4에서는 학습 후의 DQN 에이전트의 멀티캐스트 라우팅 트리 생성 과정을 도식화한 것이다. 앞서 언급한 그림 2의 과정과 같이 DQN 에이전트는 SDN 제어기로부터 네트워크 상황과 멀티캐스트 정보들을 받는다. DQN 에이전트가 멀티캐스트 라우팅 트리를 생성하는 첫 단계에서는 SDN 제어기로부터 주어진 정보들을 학습된 에이전트 신경망의 입력으로 사용하고 신경망의 결과 값으로부터 행동을 선택하며 트리 생성을 시작한다. 두 번째 단계에서는 에이전트 신경망에게 현재 연결된 링크 토폴로지와 멀티캐스트 정보를 입력으로 사용한다. 따라서 에이전트 신경망은 현재 연결된 링크 토폴로지 정보를 고려한 결과 값을 출력하고 모든 목적지를 연결할 때까지 반복한다. 모든 목적지를 연결하게 되면 멀티캐스트 라우팅 트리 구성이 완료된다. 이러한 생성 방법이 기존 연구와 다른 점은 에이전트가 학습 과정에서 멀티캐스트 라우팅 트리를 매번 생성할 때마다 새로운 환경에 노출 된다는 것이다.

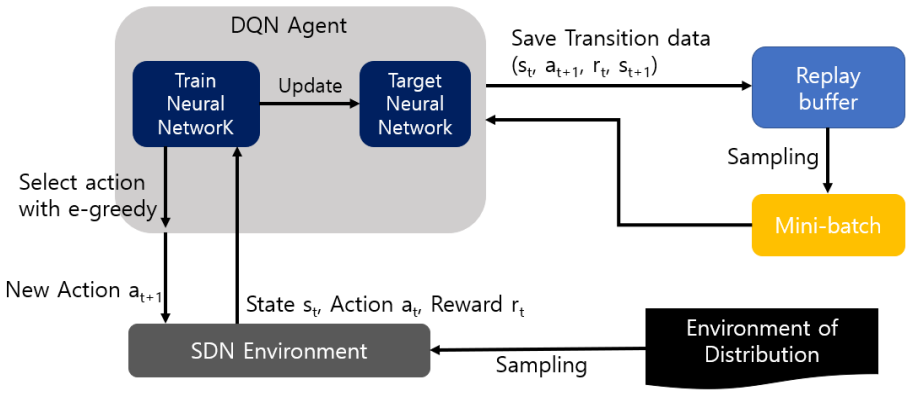


그림 3. DQN 에이전트 학습 과정
Fig. 3. Proposed process of training DQN agent

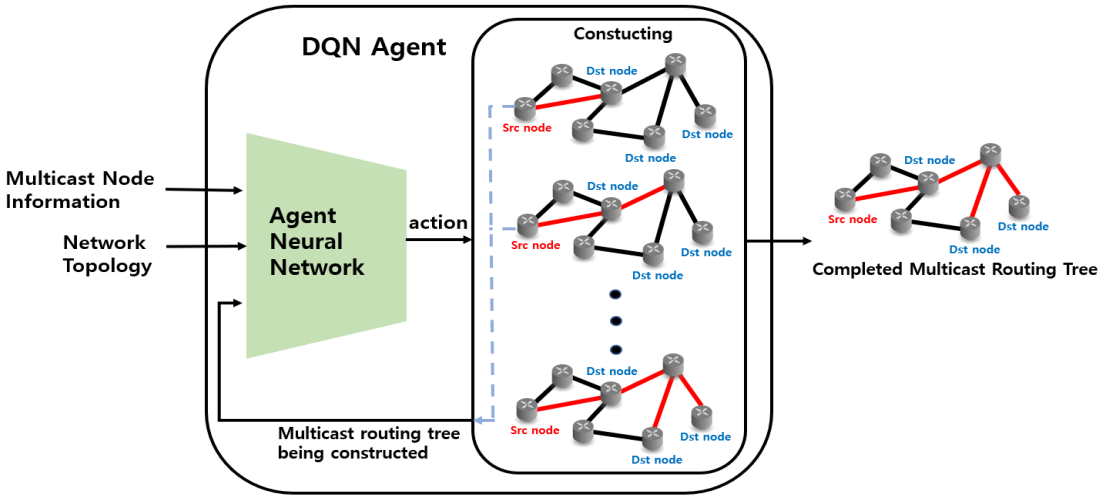


그림 4. DQN 에이전트의 멀티캐스트 라우팅 트리 생성 과정
Fig. 4. Process of constructing multicast routing tree for DQN agent

기존 연구에서는 학습 과정에서 항상 같은 환경에 대해서 멀티캐스트 라우팅 트리를 생성하므로 멀티캐스트 라우팅 트리를 생성하는 신경망의 대응 능력이 떨어지지만, 매번 새로운 환경에 처하게 되는 제안 방법은 새로운 환경에 대해 대응하여 학습하게 된다.

위와 같이 생성된 멀티캐스트 라우팅 트리의 성능을 평가하기 위해 다음 절인 실험에서는 본 논문에서 제안한 한 번의 에피소드가 종료되었을 때 새로운 환경을 추출하는 방식으로 학습한 DQN 에이전트가 일반적인 고정적 네트워크 토폴로지에서 학습한 DQN 에이전트에 비해 가변적인 네트워크 환경에서 얼마나 나은 성능을 내는지 정량적으로 평가해 본다.

Ⅳ. 실 험

4.1 실험 환경

본 논문에서 제안하는 가변적인 네트워크 링크 토폴로지를 고려한 DQN 에이전트 기반의 멀티캐스트 라우팅 트리 생성 방법을 구현하기 위해서 먼저 표 1과 표 2와 같이 데이터 집합을 생성하였다. 표 1은 고정적 네트워크 환경에서 학습을 진행하는 고정(Fixed) DQN 에이전트와 가변적 네트워크 환경에서 학습을 진행하는 범용(General) DQN 에이전트가 공통으로 사용하는 멀티캐스트 노드 집합을 나타낸다. 네트워크 구성 노드가 10개인 경우 학습에는 4,010개, 테스트에는 1,010개의 멀티캐스트 노드 집합을 사용하고 노드가 20개인 경우 학습에서는 80,000개, 테스트에는 20,000개의 노드 집합을 사용한다.

표 2는 범용(General) DQN 에이전트가 사용할 네트워크 링크 토폴로지의 수를 보여주고 있다. 이때 생성 가능한 모든 네트워크 링크 토폴로지를 사용하는 것은 현실적으로 어렵기 때문에 일부 링크 토폴로지만을 임의로 선택하여 학습에 사용한다. 그 수는 네트워크 구성 노드가 10개 일 때 50,000개, 구성 노드가 20개일 때, 100,000개이다.

표 1. DQN 에이전트 학습과 테스트를 위한 노드 데이터 개수
Table 1. Number of node data for training and testing DQN agents

Node	Total Data	Training	Testing
10	5,020	4,010	1,010
20	100,000	80,000	20,000

표 2. 범용 DQN 에이전트가 학습에 사용하는 네트워크 링크 토폴로지 개수
Table 2. Number of network link topologies for training the general DQN agents

Node	Network Link Topology
10	50,000
20	100,000

또한 실험에 사용한 노드 개수는 관련 연구와 비슷한 노드 개수를 가정하였다. 관련 연구인 [11][12]에서는 sprint network[13]를 사용하였는데 노드 개수 25개, 링크 개수 56개로 이루어져 있다. 본 논문에서 실험한 노드 개수가 비교적 작지만 다양한 링크 토폴로지에 대해서 실험하므로 적절하게 실험을 구성했다고 할 수 있다.

본 논문에서 학습한 DQN 에이전트들을 테스트하기 위해 그림 5와 같은 방법을 사용하였다. 그림에서 알 수 있듯이 각각 3개의 밀집도를 가지는 고정적 네트워크 환경에서 학습한 3개의 고정 DQN 에이전트와 가변적 네트워크에서 학습한 1개의 범용 DQN 에이전트가 존재한다. 고정 DQN 에이전트는 학습하지 않았던 다른 밀집도를 가지는 네트워크 토폴로지를 포함하여 테스트를 진행한다. 따라서 하나의 고정 DQN 에이전트마다 총 3개의 네트워크 토폴로지(학습 토폴로지 1개, 학습하지 않은 토폴로지 2개)에서 테스트 된다. 또한 범용 DQN 에이전트는 고정 DQN 에이전트와의 비교를 위해 고정 DQN 에이전트가 테스트된 네트워크 토폴로지에서 테스트를 진행한다. 그러므로 범용 DQN 에이전트는 총 3개의 네트워크 토폴로지에서 테스트 된다. 이에 따라 고정 에이전트는 자신이 학습한 네트워크 토폴로지에서는 좋은 성능을 보일 수 있으나 그 외 네트워크 토폴로지에서는 나쁜 성능을 보인다. 이에 반해 범용 에이전트는 여러 밀집도를 가지는 다양한 네트워크 토폴로지에서 고른 성능을 보인다.

학습된 DQN 에이전트의 정량적인 성능을 측정하기 위해 식 (1)과 같은 성능 비율(Performance ratio)을 사용한다. 성능 비율은 DQN 에이전트가 생성한 멀티캐스트 라우팅 트리 길이를 최적 멀티캐스트 라우팅 트리 길이로 나눈 것이다. 성능 비율 값이 1에 가까워질수록 생성된 멀티캐스트 라우팅 트리가 최적 멀티캐스트 라우팅 트리 길이에 가깝다고 말할 수 있다.

$$Performance\ ratio = \frac{Constructed\ Multicast\ Tree\ Length}{Optimal\ Multicast\ Tree\ Length} \quad (1)$$

4.2 실험 결과

학습된 DQN 에이전트들의 멀티캐스트 라우팅 트리 생성 실험 결과는 그림 6, 그림 7과 같다. 그림 5와 그림 6은 노드 개수가 각각 10개, 20개 일 때의 각 DQN 에이전트의 성능 비율을 나타낸 그래프이다. 그림에서 Fixed_30, Fixed_50, Fixed_70은 학습에 사용된 네트워크 링크 밀집도가 30%, 50%, 70%인 고정 DQN 에이전트들을 나타낸다. 결과에서 알 수 있듯이 고정 에이전트는 자신이 학습한 네트워크 링크 밀집도를 가지는 토폴로지에서는 최적 성능인 1에 근접한 성능을 보이지만 다른 밀집도를 가지는 토폴로지에서는 성능이 급격히 떨어진다. 반면에 범용 에이전트는 특정 밀집도에서는 해당 밀집도 네트워크 링크 토폴로지를 집중적으로 학습한 고정 에이전트보다는 성능이 떨어지지만 다른 링크 밀집도를 가지는 네트워크 환경에서 전반적으로 고정 에이전트보다 모두 나은 성능을 보인다.

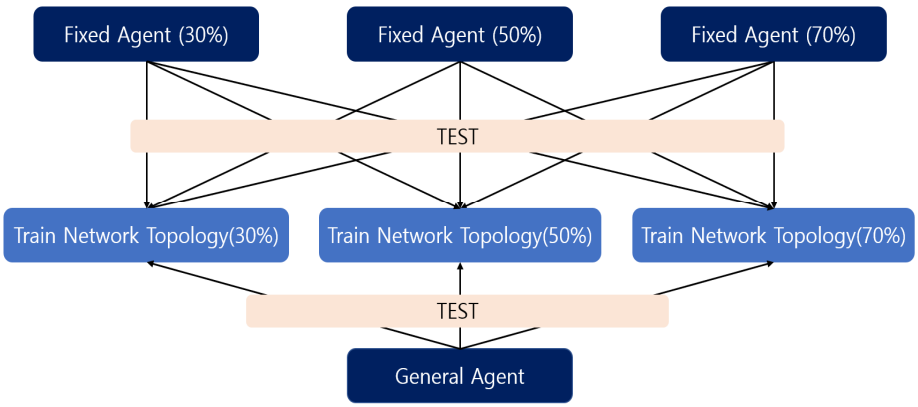


그림 5. 비교 실험 구성
Fig. 5. Comparative experiment diagram

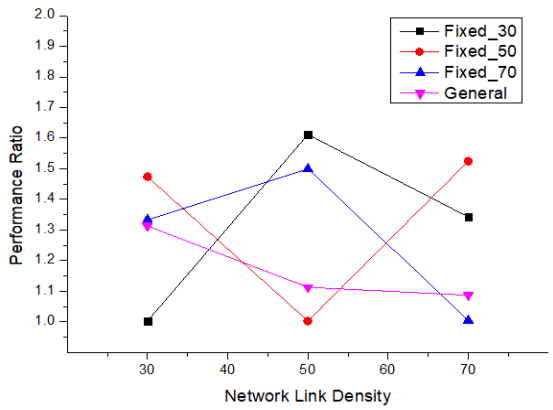


그림 6. 노드 개수가 10개일 때 각 에이전트의 성능 비율
Fig. 6. Performance ratio for each agents that the number of nodes is 10

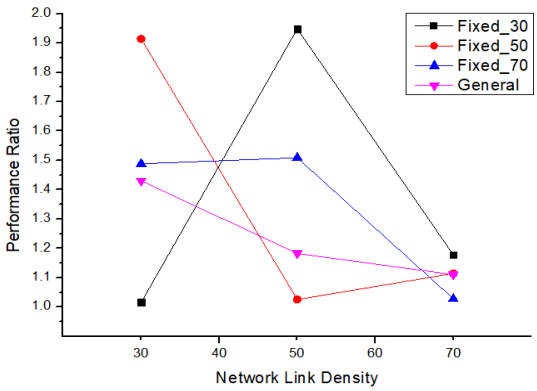


그림 7. 노드 개수가 20개일 때 각 에이전트의 성능 비율
Fig. 7. Performance ratio for each agents that the number of nodes is 20

V. 결론 및 향후 연구

하나의 네트워크 환경을 가정하고 그 환경만을 고정적으로 학습한 DQN 에이전트는 다른 네트워크 환경에서 효율적으로 멀티캐스트 라우팅 트리를 생성하지 못한다. 따라서 실제 네트워크 환경과 유사하게 시간이 지남에 따라 가변적으로 변화하는 네트워크 환경에서도 운영 가능한 범용 DQN 에이전트를 강화학습을 통해 구현할 필요가 있다. 따라서 본 논문에서는 기존의 강화학습 방법들이 고정적 환경을 가정하고 있기 때문에 발생하는 일반화 문제를 해결하기 위해 다양한 네트워크 환경을 모두 학습 할 수 있는 강화학습 방법을 제안했다. 그리고 이 방법을 SDN 환경에서 멀티캐스트 라우팅 트리 생성 문제에 적용하였다. 다음으로 다양한 노드와 밀집도를 가지는 네트워크 토폴로지에서 실험을 통해 성능을 비교하였다. 그 결과 제안한 방법이 고정적 환경을 가정한 일반적인 강화학습 방법보다 더 효과적임을 알 수 있었다.

실험 결과 가변적 네트워크 환경에서 학습한 범용 에이전트는 밀집도가 낮은 네트워크 링크 토폴로지에서 비교적 나쁜 성능을 보인다. 이를 개선하기 위해 향후 연구에서는 밀집도가 더 낮은 네트워크 링크 토폴로지의 학습데이터 비중을 늘리고 DQN과 다른 여러 강화학습 기법들을 적용해 볼 예정이다. 또한 논문에서 실험한 환경과 달리 SDN 환경에서 링크의 대역폭에 대한 추가적인 조건 등을 고려하여 보다 복잡한 네트워크 상황에서의 효율적인 멀티캐스트 라우팅 방법에 대한 연구를 진행해 볼 예정이다.

References

[1] Xie, Junfeng, et al., "A survey of machine learning techniques applied to software defined networking(SDN): Research issues and challenges", IEEE Communications Surveys & Tutorials Vol. 21, No. 1, pp. 393-430, Aug. 2018.

[2] M. Latah and T. Levent, "Application of artificial intelligence to software defined networking: A survey", Indian Journal of Science and Technology Vol. 9, No. 44, pp. 1-7, May 2016.

[3] Lin, Shih-Chun, et al., "QoS-aware adaptive routing in multi-layer hierarchical software defined networks: A reinforcement learning approach", 2016 IEEE International Conference on Services Computing (SCC), IEEE, pp. 25-33, Jun. 2016.

[4] Sun, Meng, et al., "A multiple multicast tree optimization solution based on software defined network.“, 7th Int. Conf. Inform. Commun. Syst. (ICICS). IEEE, pp. 168-173, Apr. 2016.

[5] P. Winter, "Steiner problem in networks: a survey", Networks, Vol. 17, No. 2, pp. 129-167, 1987.

[6] F. K. Hwang and D. S. Richards, "Steiner tree problems", Networks, Vol. 22, No. 1, pp. 55-89,

1992.

[7] W. Christopher JCH, and D. Peter, "Q-learning" Machine learning, Vol. 8, No. 3-4, pp. 279-292, 1992.

[8] R. S. Sutton and A. G. Barto, "Introduction to reinforcement learning", Cambridge MIT press, Vol. 135, 1998.

[9] Mnih, Volodymyr, et al., "Human-level control through deep reinforcement learning", Nature, Vol. 518, No. 7540, pp. 529-533, 2015.

[10] H. Heo, N. Kim, and B. Lee, "Multicast Tree Generation Technique Using Reinforcement Learning in SDN Environments", 2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation(SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI), IEEE, pp. 77-81, Oct. 2018.

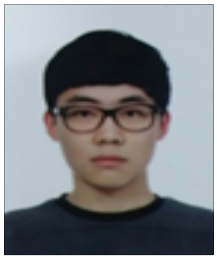
[11] Yu. C., Lan. J., Guo. Z, and Hu. Y, "DROM: Optimizing the Routing in Software-Defined Networks With Deep Reinforcement Learning", IEEE Access, Vol. 6, pp. 64533-64539, Oct. 2018.

[12] Lin, S. C., Wang, P., Akyildiz, I. F., and Luo, M., "Towards optimal network planning for software-defined networks", IEEE Transactions on Mobile Computing, Vol. 17, No. 12, pp. 2953-2967, Mar. 2018.

[13] Sprint, Overland Park, KS, USA. (2011). Sprint IP Network Performance.

저자소개

채 지 훈 (Jihun Chae)



2019년 2월 : 경기대학교 컴퓨터과학과
2019년 3월 ~ 현재 : 경기대학교 컴퓨터공학부 석사과정
관심분야 : 네트워크, 강화학습, 머신러닝

이 병 대 (Byoung-Dai Lee)

1996년 : 연세대학교 전산학과
1998년 : 연세대학교 대학원 전산학과(공학석사)
2003년 : Univ. of Minnesota CSE(공학박사)
2003년 3월 ~ 2010년 2월 : 삼성전자 책임연구원
2010년 3월 ~ 현재 : 경기대학교 컴퓨터공학부 교수
관심분야 : 기계학습, 딥러닝, 의료 영상 분석

김 남 기 (Namgi Kim)

1997년 : 서강대학교 컴퓨터학과
2000년 : KAIST 전산학과 공학석사
2005년 : KAIST 전산학과 공학박사
2005년 ~ 2007년 : 삼성전자 통신연구소 책임연구원
2007년 3월 ~ 현재 : 경기대학교 컴퓨터공학부 교수
관심분야 : 통신시스템, 네트워크